FAT-Pointer based range TLB

Akilan Selvacoumar[1], Robert Stewart[1], Hans-Wolfgang Loidl[1], and Ryad Soobhany[1]

[1]Heriot-Watt University, UK

Abstract

Capability-based addressing in the CHERI architecture is designed to improve hardware-level system security. These security guarantees can degrade runtime performance due to increased L1 TLB misses [**1**]. Our aim is to achieve capability-based security at zero performance cost. We will use fat pointers to simplify and implement a recently proposed range TLB scheme [**2**], with the goal of reducing L1 TLB misses and page-walks for memory-intensive CHERI workloads.

Research Questions

Research Questions:

- (RQ1) Can L1 misses and page-walks be reduced for memory intensive CHERI workloads?
- (RQ2) Is it possible to implement a simple Range TLB scheme with 8 bits using region of a fat pointer?
- (RQ3) Can fat pointer based address translation lower the runtime overheads of hardware-level security in capability hardware like CHERI?

Our Approach

Range translation is the process of mapping between contigous virtual pages mapped to contigous phyiscal pages [**3**]. Recent work on FlexPointer[**2**] extended RMM[**3**] by encoding RangeIDs to the remaining 16 bits on a 64-bit virtual address. In the absence of fat pointers, their Range TLB was its own data structure in memory. We use fat pointers to simplify the implementation of Range TLB by using the bounds metadata to extract the lower and upper bounds which inturns allows us to encode the offset to the Pointer to map to the Physical page number. We hope to reproduce the FlexPointer results of reducing L1 tlb misses and page walks.

References

References

[1] Bramley, J., Jacob, D., Lascu, A., Singer, J. & Tratt, L. Picking a CHERI Allocator: Security and Performance Considerations. *Proceedings Of The 2023 ACM SIGPLAN International Symposium On Memory Management*. (2023,6), https://doi.org/10.1145

[2] Chen, D., Tong, D., Yang, C., Yi, J. & Cheng, X. FlexPointer: Fast Address Translation Based on Range TLB and Tagged Pointers. *ACM Trans. Archit. Code Optim.*. **20** (2023,3), https://doi.org/10.1145/3579854

[3] Karakostas, V., Gandhi, J., Ayar, F., Cristal, A., Hill, M., McKinley, K., Nemirovsky, M., Swift, M. & Ünsal, O. Redundant Memory Mappings for Fast Access to Large Memories. *SIGARCH Comput. Archit. News*. **43**, 66-78 (2015,6), https://doi.org/10.1145/2872887.2749471

Architecture